EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S1	27	cyclic near3 garbage	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/01/08 14:32
S2	10	(cyclic near3 garbage) same (time timestamp) same count same object	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/01/08 14:40
S3	144	(garbage near5 collect$5) same (time timestamp) same lifetime same object	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/01/08 16:24
S4	80	S3 and @ad<"20020403"	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/01/08 14:42
S5	119	(garbage near5 collect$5) same (time timestamp) same (lifetime near3 object)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/01/08 16:25
S6	64	S5 and @ad<"20020403"	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/01/08 14:47
S7	3	(garbage near5 collect$5) same (time timestamp) same (lifetime near3 object) same dead	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/01/08 14:43
S8	24	(garbage adj collection) same (object near5 (time timestamp cyclic)) same (death dead)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2008/01/08 16:25

S9	6	(garbage adj collection) same (object same ((time timestamp cyclic) near5 (death dead)))	US-PGPUB;
			USPAT;
			USOCR;
			FPRS;
			EPO; JPO;
			DERWENT;
			IBM_TDB
				OR	ON	2008/01/08 14:50

S10	16	(garbage adj collection) same (object near5 (time timestamp cyclic)) same (death dead) and "707"/$.ccls.	US-PGPUB;
			USPAT;
			USOCR;
			FPRS;
			EPO; JPO;
			DERWENT;
			IBM_TDB
				OR	ON	2008/01/08 14:55

S11	13	(maintain$3 same count same objects same pointers same object).clm.	US-PGPUB;
			USPAT;
			USOCR;
			FPRS;
			EPO; JPO;
			DERWENT;
			IBM_TDB
				OR	ON	2008/01/08 14:56

S12	2	(record$3 same timestamp same object same reference same count same changes).clm.	US-PGPUB;
			USPAT;
			USOCR;
			FPRS;
			EPO; JPO;
			DERWENT;
			IBM_TDB
				OR	ON	2008/01/08 14:57

S13	2	(review$3 same reverse same chronological same order same timestamps same object same cyclic near garbage).clm.	US-PGPUB;
			USPAT;
			USOCR;
			FPRS;
			EPO; JPO;
			DERWENT;
			IBM_TDB
				OR	ON	2008/01/08 14:58

S14	2	(indicat$3 same object same reachable same object same timestamp same dead).clm.	US-PGPUB;
			USPAT;
			USOCR;
			FPRS;
			EPO; JPO;
			DERWENT;
			IBM_TDB
				OR	ON	2008/01/08 14:59

S15	40	(garbage near5 collect$5) same (time timestamp) same lifetime same object	US-PGPUB	OR	ON	2008/01/08 16:25
S16	28	(garbage near5 collect$5) same (time timestamp) same (lifetime near3 object)	US-PGPUB	OR	ON	2008/01/08 16:38
S17	9	(garbage adj collection) same (object near5 (time timestamp cyclic)) same (death dead)	US-PGPUB	OR	ON	2008/01/08 16:37

PALM INTRANET

Day : Tuesday
Date: 1/8/2008
Time: 17:46:46

Inventor Name Search Result

Your Search was:

Last Name = WOLCZKO
First Name = MARIO

Application#	Patent#	Status	Date Filed	Title	Inventor Name
09107382	Not Issued	161	06/30/1998	FEEDBACK-BASED MEMORY ALLOCATION OPTIMIZATION IN A GARBAGE COLLECTION MEMORY MANAGEMENT SCHEME	WOLCZKO, MARIO
09229272	6842853	150	01/13/1999	THREAD SUSPENSION SYSTEM AND METHOD	WOLCZKO, MARIO
09255226	6308319	150	02/22/1999	THREAD SUSPENSION SYSTEM AND METHOD USING TRAPPING INSTRUCTIONS IN DELAY SLOTS	WOLCZKO, MARIO
09363283	Not Issued	161	07/28/1999	SINGLE-COMPILER ARCHITECTURE	WOLCZKO, MARIO
09466335	Not Issued	161	12/17/1999	METHOD AND APPARATUS FOR MONITORING A CACHE FOR GARBAGE COLLECTION	WOLCZKO, MARIO
09910423	Not Issued	160	07/20/2001	Thread suspension system and method using trapping instructions	WOLCZKO, MARIO
09986231	7013454	150	10/22/2001	THREAD SUSPENSION SYSTEM AND METHOD USING TRAPPING INSTRUCTIONS	WOLCZKO, MARIO
10113357	7096390	150	04/01/2002	SAMPLING MECHANISM INCLUDING INSTRUCTION FILTERING	WOLCZKO, MARIO
10116236	6728738	150	04/03/2002	FAST LIFETIME ANALYSIS OF OBJECTS IN A GARBAGE-COLLECTED SYSTEM	WOLCZKO, MARIO
10796539	Not Issued	71	03/08/2004	Fast lifetime analysis of objects in a garbage collected system	WOLCZKO, MARIO
10861704	Not Issued	89	06/04/2004	Thread suspension system and method	WOLCZKO, MARIO
11373949	Not Issued	30	03/13/2006	Cooperative preemption mechanism for garbage-collected multi-threaded computation	WOLCZKO, MARIO
11648135	Not Issued	30	12/28/2006	Methods and apparatus for marking objects for garbage collection in an object-based memory system	WOLCZKO, MARIO

60253551	Not Issued	159	11/28/2000	Portable and automated native code isolation	WOLCZKO, MARIO
09675116	6993761	150	09/28/2000	METHOD AND APPARATUS TO VERIFY TYPE SAFETY OF AN APPLICATION SNAPSHOT	WOLCZKO, MARIO I.
09841719	6834391	150	04/24/2001	METHOD AND APPARATUS FOR AUTOMATED NATIVE CODE ISOLATION	WOLCZKO, MARIO I.
09992063	6745213	150	11/21/2001	METHOD AND APPARATUS TO FACILITATE TESTING OF GARBAGE COLLECTION IMPLEMENTATIONS	WOLCZKO, MARIO I.
10072169	6859868	150	02/07/2002	OBJECT ADDRESSED MEMORY HIERARCHY	WOLCZKO, MARIO I.
10124122	6950838	150	04/17/2002	LOCATING REFERENCES AND ROOTS FOR IN-CACHE GARBAGE COLLECTION	WOLCZKO, MARIO I.
10146268	6751709	150	05/15/2002	METHOD AND APPARATUS FOR PREFETCHING OBJECTS INTO AN OBJECT CACHE	WOLCZKO, MARIO I.
10222613	7036112	150	08/16/2002	MULTI-MODE SPECIFICATION-DRIVEN DISASSEMBLER	WOLCZKO, MARIO I.
10335621	7246141	150	01/02/2003	METHOD AND APPARATUS FOR SKEWING A BI-DIRECTIONAL OBJECT LAYOUT TO IMPROVE CACHE PERFORMANCE	WOLCZKO, MARIO I.
10389629	6934827	150	03/13/2003	METHOD AND APPARATUS FOR AVOIDING CACHE LINE COLLISIONS BETWEEN AN OBJECT AND CORRESPONDING OBJECT TABLE ENTRIES	WOLCZKO, MARIO I.
10431116	6931504	150	05/06/2003	METHOD AND APPARATUS FOR RELOCATING OBJECTS WITHIN AN OBJECT-ADDRESSED MEMORY HIERARCHY	WOLCZKO, MARIO I.
10646309	Not Issued	61	08/22/2003	Reducing the overhead involved in executing native code in a virtual machine through binary reoptimization	WOLCZKO, MARIO I.
10698727	7249225	150	10/31/2003	METHOD AND APPARATUS FOR SUPPORTING READ-ONLY OBJECTS WITHIN AN OBJECT-ADDRESSED MEMORY HIERARCHY	WOLCZKO, MARIO I.
10698728	7171540	150	10/31/2003	OBJECT-ADDRESSED MEMORY HIERARCHY THAT FACILITATES ACCESSING OBJECTS STORED OUTSIDE OF MAIN MEMORY	WOLCZKO, MARIO I.
10779216	Not Issued	41	02/13/2004	Performance counters in a multi-threaded processor	WOLCZKO, MARIO I.
10780264	Not Issued	71	02/16/2004	Instruction sampling in a multi-threaded processor	WOLCZKO, MARIO I.
10784730	Not Issued	83	02/23/2004	Obtaining execution path information in an instruction	WOLCZKO, MARIO I.

				sampling system by linking control transfer information with sampling information	
10792441	Not Issued	61	03/03/2004	Incorporating instruction reissue in an instruction sampling mechanism with persistent and resettable sample information	WOLCZKO, MARIO I.
10831415	7269705	150	04/23/2004	Memory space management for object-based memory system	WOLCZKO, MARIO I.
10838309	7167956	150	05/03/2004	AVOIDING INCONSISTENCIES BETWEEN MULTIPLE TRANSLATORS IN AN OBJECT-ADDRESSED MEMORY HIERARCHY	WOLCZKO, MARIO I.
10849081	Not Issued	41	05/18/2004	Method and system for concurrent garbage collection and mutator execution	WOLCZKO, MARIO I.
10878866	Not Issued	161	06/28/2004	System for identifying instructions for sampling	WOLCZKO, MARIO I.
10880485	Not Issued	41	06/30/2004	Associating data source information with runtime events	WOLCZKO, MARIO I.
10893090	Not Issued	95	07/16/2004	METHOD FOR MONITORING HEAP FOR MEMORY LEAKS	WOLCZKO, MARIO I.
10903169	Not Issued	41	07/29/2004	Method and apparatus for maintaining an object-based write barrier to facilitate garbage-collection operations	WOLCZKO, MARIO I.
10944172	Not Issued	94	09/16/2004	METHOD AND APPARATUS FOR USING MEMORY COMPRESSION TO ENHANCE ERROR CORRECTION	WOLCZKO, MARIO I.
11042688	Not Issued	41	01/24/2005	Method and apparatus for generating an execution profile for an application	WOLCZKO, MARIO I.
11155276	Not Issued	80	06/17/2005	Method and apparatus for facilitating in-cache reference counting	WOLCZKO, MARIO I.
11325381	Not Issued	30	01/03/2006	Method and apparatus for facilitating mark-sweep garbage collection with reference counting	WOLCZKO, MARIO I.
11325383	Not Issued	30	01/03/2006	Memory management system that supports both address-referenced objects and identifier-referenced objects	WOLCZKO, MARIO I.
11346399	Not Issued	30	02/01/2006	Multiprocessor system that supports both coherent and non-coherent memory accesses	WOLCZKO, MARIO I.
11516972	Not Issued	30	09/07/2006	Method and system for extending evaluation for intermediate representation interpretation	WOLCZKO, MARIO I.
11864847	Not Issued	17	09/28/2007	METHOD AND SYSTEM FOR IMPLEMENTING A JUST-IN-TIME COMPILER	WOLCZKO, MARIO I.

60518471	Not Issued	159	11/07/2003	Object-aware memory architecture	WOLCZKO, MARIO I.
60564035	Not Issued	159	04/21/2004	Associating data source information with runtime events	WOLCZKO, MARIO I.
08838958	5920876	150	04/23/1997	PERFORMING EXACT GARBAGE COLLECTION USING BITMAPS THAT IDENTIFY POINTER VALUES WITHIN	WOLCZKO, MARIO I.
08842136	6115782	150	04/23/1997	METHOD AND APPARATUS FOR LOCATING NODES IN A CARDED HEAP USING A CARD MARKING STRUCTURE AND A NODE ADVANCE VALUE	WOLCZKO, MARIO I.

Search and Display More Records.

Search Another: Inventor

Last Name WOLCZKO

First Name MARIO

Search

To go back use Back button on your browser toolbar.

Back to PALM | ASSIGNMENT | OASIS | Home page

PALM INTRANET

Day : Tuesday
Date: 1/8/2008
Time: 17:47:10

Inventor Name Search Result

Your Search was:

Last Name = CUNEI
First Name = ANTONIO

Application#	Patent#	Status	Date Filed	Title	Inventor Name
10116236	6728738	150	04/03/2002	FAST LIFETIME ANALYSIS OF OBJECTS IN A GARBAGE-COLLECTED SYSTEM	CUNEI, ANTONIO
10796539	Not Issued	71	03/08/2004	Fast lifetime analysis of objects in a garbage collected system	CUNEI, ANTONIO
11011734	7191307	150	12/14/2004	MEMORY MANAGEMENT UNIT TECHNIQUE TO DETECT CROSS-REGION POINTER STORES	CUNEI, ANTONIO

Inventor Search Completed: No Records to Display.

	Last Name	First Name	
Search Another: Inventor	CUNEI	ANTONIO	Search

To go back use Back button on your browser toolbar.

Back to PALM | ASSIGNMENT | OASIS | Home page

Web Images Maps News Shopping Gmail more ▾ Sign in

Google

"garbage collector" "dead timestamp" lifetime (Search Advanced Search Preferences

Web Results **1** - **3** of **3** for "**garbage collector**" "**dead timestamp**" **lifetime object**. (**0.19** seconds)

Fast **lifetime** analysis of **objects** in a garbage-collected system ...
An apparatus for measuring the **lifetime** of **objects** in a garbage-collected ... having a **dead timestamp object** indicator, a dead reachable **object** indicator, ...
www.patentstorm.us/patents/6728738-claims.html - 27k - Cached - Similar pages

Fast **lifetime** analysis of **objects** in a garbage-collected system ...
A **garbage collector** 710 coupled to the memory 700 executes a garbage collection, ... A dead **object** cyclic garbage remover 718 coupled to the **dead timestamp** ...
www.patentstorm.us/patents/6728738-description.html - 37k - Cached - Similar pages

Fast **lifetime** analysis of **objects** in a garbage-collected system ...
The apparatus of claim 18, wherein said **garbage collector** includes a tracing ..... A dead **object** cyclic garbage remover 718 coupled to the **dead timestamp** ...
www.freepatentsonline.com/6728738.html - 48k - Cached - Similar pages

In order to show you the most relevant results, we have omitted some entries very similar to the 3 already displayed. If you like, you can repeat the search with the omitted results included.

"garbage collector" "dead timestamp Search

Search within results | Language Tools | Search Tips | Dissatisfied? Help us improve | Try Google Experimental

PORTAL

USPTO

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: ○ The ACM Digital Library ● The Guide

+"garbage collector" +dead +timestamp +lifetime +object SEARCH

THE GUIDE TO COMPUTING LITERATURE

Feedback Report a problem Satisfaction survey

Terms used: garbage collector dead timestamp lifetime object

Found 20 of 1,101,485

Sort results by relevance

Display results expanded form

Save results to a Binder

Search Tips

□ Open results in a new window

Try an Advanced Search
Try this search in The Digital Library

Results 1 - 20 of 20

Relevance scale

1 Generating object lifetime traces with Merlin

Matthew Hertz, Stephen M. Blackburn, J. Eliot B. Moss, Kathryn S. McKinley, Darko Stefanović

May 2006 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 28 Issue 3

Publisher: ACM Press

Full text available: pdf(1.31 MB) Additional Information: full citation, abstract, references, index terms

Programmers are writing a rapidly growing number of programs in object-oriented languages, such as Java and C&num;, that require garbage collection. Garbage collection traces and simulation speed up research by enabling deeper understandings of object lifetime behavior and quick exploration and design of new garbage collection algorithms. When generating perfect traces, the *brute-force* method of computing object lifetimes requires a whole-heap garbage collection at every potential collect ...

Keywords: Garbage collection, object lifetime analysis, trace design, trace generation

2 Error-free garbage collection traces: how to cheat and not get caught

Matthew Hertz, Stephen M Blackburn, J Eliot B Moss, Kathryn S. McKinley, Darko Stefanović

June 2002 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 2002 ACM SIGMETRICS international conference on Measurement and modeling of computer systems SIGMETRICS '02**, Volume 30 Issue 1

Publisher: ACM Press

Full text available: pdf(105.06 KB) Additional Information: full citation, abstract, references, citings

Programmers are writing a large and rapidly growing number of programs in object-oriented languages such as Java that require garbage collection (GC). To explore the design and evaluation of GC algorithms quickly, researchers are using simulation based on traces of object allocation and lifetime behavior. The *brute force* method generates perfect traces using a whole-heap GC at every potential GC point in the program. Because this process is prohibitively expensive, researchers often use < ...

3 Incremental distribution of timestamp packets: a new approach to distributed garbage collection

M. Schelvis

September 1989 **ACM SIGPLAN Notices , Conference proceedings on Object-oriented programming systems, languages and applications OOPSLA '89**, Volume 24 Issue 10

Publisher: ACM Press

Full text available: pdf(1.20 MB) Additional Information: full citation, abstract, references, citings, index terms

A new algorithm for distributed garbage collection is presented. This algorithm collects distributed garbage incrementally and concurrently with user activity. It is the first incremental algorithm that is capable of collecting cyclic distributed garbage. Computational and network communication overhead are acceptable. Hosts may be temporarily inaccessible and synchronization between hosts is not necessary. The algorithm is based on asynchronous distribution of timestamp pa ...

4 Garbage collecting the Internet: a survey of distributed garbage collection

Saleh E. Abdullahi, Graem A. Ringwood
September 1998 **ACM Computing Surveys (CSUR)**, Volume 30 Issue 3
Publisher: ACM Press
Full text available: pdf(337.65 KB) Additional Information: full citation, abstract, references, citings, index terms, review

Internet programming languages such as Java present new challenges to garbage-collection design. The spectrum of garbage-collection schema for linked structures distributed over a network are reviewed here. Distributed garbage collectors are classified first because they evolved from single-address-space collectors. This taxonomy is used as a framework to explore distribution issues: locality of action, communication overhead and indeterministic communication latency.

Keywords: automatic storage reclamation, distributed, distributed file systems, distributed memories, distributed object-oriented management, memory management, network communication, object-oriented databases, reference counting

5 Memory system behavior of Java programs: methodology and analysis
Jin-Soo Kim, Yarsun Hsu
June 2000 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 2000 ACM SIGMETRICS international conference on Measurement and modeling of computer systems SIGMETRICS '00**, Volume 28 Issue 1
Publisher: ACM Press
Full text available: pdf(1.08 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper studies the memory system behavior of Java programs by analyzing memory reference traces of several SPECjvm98 applications running with a Just-In-Time (JIT) compiler. Trace information is collected by an exception-based tracing tool called JTRACE, without any instrumentation to the Java programs or the JIT compiler.First, we find that the overall cache miss ratio is increased due to garbage collection, which suffers from higher cache misses compared to the application. ...

6 Quantifying the performance of garbage collection vs. explicit memory management
Matthew Hertz, Emery D. Berger
October 2005 **ACM SIGPLAN Notices , Proceedings of the 20th annual ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications OOPSLA '05**, Volume 40 Issue 10
Publisher: ACM Press
Full text available: pdf(1.51 MB) Additional Information: full citation, abstract, references, citings, index terms

Garbage collection yields numerous software engineering benefits, but its quantitative impact on performance remains elusive. One can compare the cost of *conservative* garbage collection to explicit memory management in C/C++ programs by linking in an appropriate collector. This kind of direct comparison is not possible for languages designed for garbage collection (e.g., Java), because programs in these languages naturally do not contain calls to free. Thus, the actual gap between the tim ...

Keywords: explicit memory management, garbage collection, oracular memory management, paging, performance analysis, throughput, time-space tradeoff

7 Object equality profiling
Darko Marinov, Robert O'Callahan
October 2003 **ACM SIGPLAN Notices , Proceedings of the 18th annual ACM SIGPLAN conference on Object-oriented programing, systems, languages, and applications OOPSLA '03**, Volume 38 Issue 11
Publisher: ACM Press
Full text available: pdf(577.47 KB) Additional Information: full citation, abstract, references, citings, index terms

We present *Object Equality Profiling* (OEP), a new technique for helping programmers discover optimization opportunities in programs. OEP discovers opportunities for replacing a set of equivalent object instances with a single representative object. Such a set represents an opportunity for automatically or manually applying optimizations such as hash consing, heap compression, lazy allocation, object caching, invariant hoisting, and more. To evaluate OEP, we implemented a tool to help prog ...

Keywords: Java language, object equality, object mergeability, profile-guided optimization, profiling, space savings

8 Object lifetimes: Allocation-phase aware thread scheduling policies to improve garbage collection performance

Feng Xian, Witawas Srisa-an, Hong Jiang

October 2007 **Proceedings of the 6th international symposium on Memory management ISMM '07**

Publisher: ACM

Full text available: pdf(805.33 KB) Additional Information: full citation, abstract, references, index terms

Past studies have shown that objects are created and then die in phases. Thus, one way to sustain good garbage collection efficiency is to have a large enough heap to allow many allocation phases to complete and most of the objects to die before invoking garbage collection. However, such an operating environment is hard to maintain in large multithreaded applications because most typical time-sharing schedulers are not allocation-phase cognizant; i.e., they often schedule threads in a way tha ...

Keywords: garbage collection, thread scheduling

9 Robust, distributed references and acyclic garbage collection

Marc Shapiro, Peter Dickman, David Plainfossé

October 1992 **Proceedings of the eleventh annual ACM symposium on Principles of distributed computing PODC '92**

Publisher: ACM Press

Full text available: pdf(1.27 MB) Additional Information: full citation, references, citings, index terms, review

10 Reliability and security: Memory overflow protection for embedded systems using run-time checks, reuse and compression

Surupa Biswas, Matthew Simpson, Rajeev Barua

September 2004 **Proceedings of the 2004 international conference on Compilers, architecture, and synthesis for embedded systems CASES '04**

Publisher: ACM Press

Full text available: pdf(253.51 KB) Additional Information: full citation, abstract, references, citings, index terms, review

Out-of-memory errors are a serious source of unreliability in most embedded systems. Applications run out of main memory because of the frequent difficulty of estimating the memory requirement before deployment, either because it depends on input data, or because certain language features prevent estimation. The typical lack of disks and virtual memory in embedded systems has two serious consequences when an out-of-memory error occurs. First, there is no swap space for the application to grow in ...

Keywords: data compression, heap overflow, out-of-memory errors, reliability, reuse, runtime checks, stack overflow

11 Memory overflow protection for embedded systems using run-time checks, reuse, and compression

Surupa Biswas, Thomas Carley, Matthew Simpson, Bhuvan Middha, Rajeev Barua

November 2006 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 5 Issue 4

Publisher: ACM Press

Full text available: pdf(579.85 KB) Additional Information: full citation, abstract, references, index terms

Embedded systems usually lack virtual memory and are vulnerable to memory overflow since they lack a mechanism to detect overflow or use swap space thereafter. We present a method to detect memory overflows using compiler-inserted software run-time checks. Its overheads in run-time and energy are 1.35 and 1.12&percnt;, respectively. Detection of overflow allows system-specific remedial action. We also present techniques to grow the stack or heap segment after they overflow, into previously unuti ...

Keywords: Out-of-memory errors, data compression, heap overflow, reliability, reuse, run-time checks, stack overflow

12 An adaptive tenuring policy for generation scavengers

David Ungar, Frank Jackson
January 1992 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 14 Issue 1
Publisher: ACM Press
Full text available: pdf(1.74 MB) Additional Information: full citation, abstract, references, citings, index terms, review

One of the more promising automatic storage reclamation techniques, generation scavenging, suffers poor performance if many objects live for a fairly long time and then die. We have investigated the severity of this problem by simulating a two-generation scavenger using traces taken from actual 4-h sessions. There was a wide variation in the sample runs, with garbage-collection overhead ranging from insignificant, during three of the runs, to severe, during a single run. All runs demonstrat ...

13 Equal rights for functional objects or, the more things change, the more they are the same
Henry G. Baker
October 1993 **ACM SIGPLAN OOPS Messenger**, Volume 4 Issue 4
Publisher: ACM Press
Full text available: pdf(2.61 MB) Additional Information: full citation, abstract, citings, index terms

We argue that intensional *object identity* in object-oriented programming languages and databases is best defined operationally by side-effect semantics. A corollary is that "functional" objects have extensional semantics. This model of object identity, which is analogous to the normal forms of relational algebra, provides cleaner semantics for the value-transmission operations and built-in primitive equality predicate of a programming language, and eliminates the confusion surrounding "ca ...

14 Shredding your garbage: reducing data lifetime through secure deallocation
Jim Chow, Ben Pfaff, Tal Garfinkel, Mendel Rosenblum
July 2005 **Proceedings of the 14th conference on USENIX Security Symposium - Volume 14 SSYM'05**
Publisher: USENIX Association
Full text available: pdf(607.54 KB) Additional Information: full citation, abstract, references, index terms

Today's operating systems, word processors, web browsers, and other common software take no measures to promptly remove data from memory. Consequently, sensitive data, such as passwords, social security numbers, and confidential documents, often remains in memory indefinitely, significantly increasing the risk of exposure.

We present a strategy for reducing the lifetime of data in memory called secure deallocation. With secure deal-location we zero data either at deallocation or within ...

15 Garbage collecting the world
Bernard Lang, Christian Queinnec, José Piquer
February 1992 **Proceedings of the 19th ACM SIGPLAN-SIGACT symposium on Principles of programming languages POPL '92**
Publisher: ACM Press
Full text available: pdf(1.39 MB) Additional Information: full citation, abstract, references, citings, index terms

Distributed symbolic computations involve the existence of remote references allowing an object, local to a processor, to designate another object located on another processor. To reclaim inaccessible objects is the non trivial task of a distributed Garbage Collector (GC). We present in this paper a new distributed GC algorithm which (i) is fault-tolerant, (ii) is largely independent of how a processor garbage collects its own data space, < ...

16 Tenuring policies for generation-based storage reclamation
David Ungar, Frank Jackson
January 1988 **ACM SIGPLAN Notices , Conference proceedings on Object-oriented programming systems, languages and applications OOPSLA '88**, Volume 23 Issue 11
Publisher: ACM Press
Full text available: pdf(1.54 MB) Additional Information: full citation, abstract, references, citings, index terms

One of the most promising automatic storage reclamation techniques, generation-based storage reclamation, suffers poor performance if many objects live for a fairly long time and then die. We have investigated the severity of this problem by simulating Generation Scavenging automatic storage reclamation from traces of actual four-hour sessions. There was a wide variation in the sample runs, with garbage-collection overhead ranging from insignificant, during the interactive runs, to severe, ...

17 First International Workshop on Persistence and Java
Malcolm Atkinson, Mick Jordan
November 1996 Technical Report
Publisher: Sun Microsystems, Inc.
Full text available: pdf(1.54 MB) Additional Information: full citation, abstract

These proceedings record the First International Workshop on Persistence and Java, which was held in Drymen, Scotland in September 1996. The focus of this workshop was the relationship between the Java languages and long-term data storage, such as databases and orthogonal persistence. There are many approaches being taken, some pragmatic and some guided by design principles. If future application programmers building large and long-lived systems are to be well-supported, it is essential that the ...

18 Myths and realities: the performance impact of garbage collection
Stephen M. Blackburn, Perry Cheng, Kathryn S. McKinley
June 2004 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the joint international conference on Measurement and modeling of computer systems SIGMETRICS '04/Performance '04**, Volume 32 Issue 1
Publisher: ACM Press
Full text available: pdf(305.06 KB) Additional Information: full citation, abstract, references, citings, index terms, review

This paper explores and quantifies garbage collection behavior for three whole heap collectors and generational counterparts: *copying semi-space, mark-sweep,* and *reference counting*, the canonical algorithms from which essentially all other collection algorithms are derived. Efficient implementations in MMTk, a Java memory management toolkit, in IBM's Jikes RVM share all common mechanisms to provide a clean experimental platform. Instrumentation separates collector and program behav ...

Keywords: generational, java, mark-sweep, reference counting, semi-space

19 Higher-order distributed objects
Henry Cejtin, Suresh Jagannathan, Richard Kelsey
September 1995 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 17 Issue 5
Publisher: ACM Press
Full text available: pdf(2.33 MB) Additional Information: full citation, abstract, references, citings, index terms, review

We describe a distributed implementation of Scheme that permits efficient transmission of higher-order objects such as closures and continuations. The integration of distributed communication facilities within a higher-order programming language engenders a number of new abstractions and paradigms for distributed computing. Among these are user-specified load-balancing and migration policies for threads, incrementally linked distributed computations, and parameterized client-server applicat ...

Keywords: concurrency, continuations, higher-order languages, message-passing

20 Languages: High-level real-time programming in Java
David F. Bacon, Perry Cheng, David Grove, Michael Hind, V. T. Rajan, Eran Yahav, Matthias Hauswirth, Christoph M. Kirsch, Daniel Spoonhower, Martin T. Vechev
September 2005 **Proceedings of the 5th ACM international conference on Embedded software EMSOFT '05**
Publisher: ACM Press
Full text available: pdf(341.79 KB) Additional Information: full citation, abstract, references, index terms

Real-time systems have reached a level of complexity beyond the scaling capability of the low-level or restricted languages traditionally used for real-time programming.While Metronome garbage collection has made it practical to use Java to implement real-time systems, many challenges remain for the construction of complex real-time systems, some specific to the use of Java and others simply due to the change in scale of such systems.The goal of our current research is the creation of a comprehe ...

Keywords: WCET, allocation, scheduling, tasks, visualization

Results 1 - 20 of 20